BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

(19) JAPANESE PATENT OFFICE

JPA2002-202865

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **2002202865 A**

(43) Date of publication of application: **19.07.02**

(51) Int. Cl **G06F 3/12**
B41J 29/38

(21) Application number: **2000401218**

(22) Date of filing: **28.12.00**

(71) Applicant: **CANON INC**

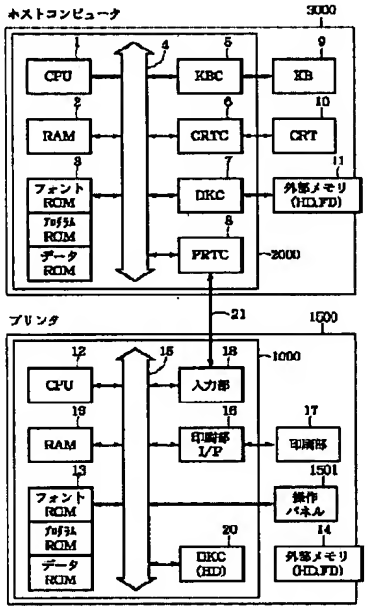
(72) Inventor: **KAWAMOTO KOICHI**

(54) **UNIT AND METHOD FOR PRINT CONTROL AND STORAGE MEDIUM**

(57) Abstract:

PROBLEM TO BE SOLVED: To actualize more accurate conflict processing while automatically generating a conflict condition to complementarily be met on the basis of a conflict condition that a program developer prepares and adding the generated condition to the conflict condition.

SOLUTION: Conflict processing rule settings obtained by putting together a plurality of conflict processing rules stored in an external memory 11, etc., are read in a RAM 2 and on the basis of the set values of the read-in conflict processing rules, a CPU 1 automatically generates a conflict processing rule to complementarily be met and uses the automatically generated conflict processing rule to perform conflict processing regarding various settings needed for the print execution to a printer 1500.



JPA 2002-202865

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2002－202865
（P2002－202865A）
(43)公開日　平成14年７月19日(2002.7.19)

(51)Int.Cl.⁷	識別記号	ＦＩ		テーマコード(参考)
Ｇ０６Ｆ　3/12		Ｇ０６Ｆ　3/12	Ｃ	２Ｃ０６１
Ｂ４１Ｊ　29/38		Ｂ４１Ｊ　29/38	Ｚ	５Ｂ０２１

審査請求　未請求　請求項の数９　ＯＬ　（全 12 頁）

(21)出願番号　特願2000－401218(P2000－401218)

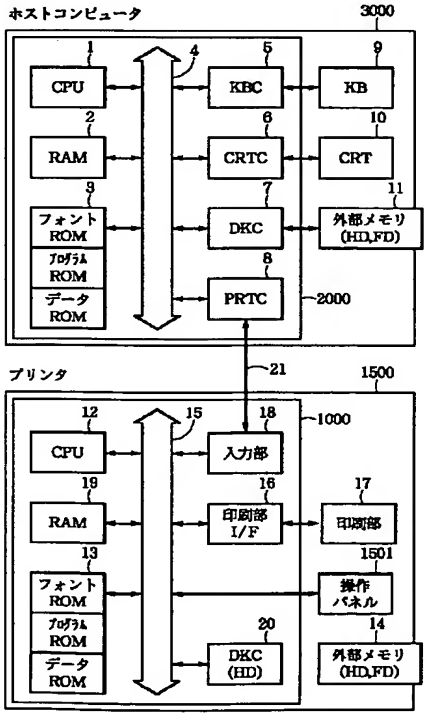
(22)出願日　平成12年12月28日(2000.12.28)

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号
(72)発明者　川本　浩一
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内
(74)代理人　100071711
弁理士　小林　将高

Ｆターム(参考)　2C061 AP01 HH03 HJ10 HK11 HN05
HN15 HN26 HP06
5B021 AA01 CC06

(54)【発明の名称】　印刷制御装置および印刷制御方法および記憶媒体

(57)【要約】
【課題】　プログラム開発者などが用意するコンフリクト条件を基に補完的に成立するコンフリクト条件を自動生成して、該コンフリクト条件に加えながら、より確かなコンフリクト処理を実現することである。
【解決手段】　外部メモリ１１等に記憶された複数のコンフリクト処理ルールをまとめたコンフリクト処理ルール設定をＲＡＭ２上に読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、ＣＰＵ１が補完的に成立するコンフリクト処理ルールを自動生成し、該自動生成されたコンフリクト処理ルールを用いて、プリンタ１５００に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行する構成を特徴とする。



【特許請求の範囲】

【請求項１】　アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置であって、

独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を記憶する記憶手段と、

前記記憶手段に記憶された複数のコンフリクト処理ルールをまとめたコンフリクト処理ルール設定を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成手段と、

前記生成手段により自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理手段と、を備えることを特徴とする印刷制御装置。

【請求項２】　前記コンフリクト処理手段によりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知手段を備えることを特徴とする請求項１記載の印刷制御装置。

【請求項３】　前記印刷処理関連プログラムは、プリンタドライバであることを特徴とする請求項１記載の印刷制御装置。

【請求項４】　アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置における印刷制御方法であって、

記憶手段に記憶される独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成ステップと、

前記生成ステップにより自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理ステップと、を備えることを特徴とする印刷制御方法。

【請求項５】　前記コンフリクト処理ステップによりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知ステップを備えることを特徴とする請求項４記載の印刷制御方法。

【請求項６】　前記印刷処理関連プログラムは、プリンタドライバであることを特徴とする請求項４記載の印刷制御方法。

【請求項７】　アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置に、

記憶手段に記憶される独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成ステップと、

前記生成ステップにより自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理ステップとを実行させるためのプログラムを記録したコンピュータが読み取り可能な記憶媒体。

【請求項８】　前記コンフリクト処理ステップによりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知ステップを備えることを特徴とする請求項７記載の記憶媒体。

【請求項９】　前記印刷処理関連プログラムは、プリンタドライバであることを特徴とする請求項７記載の記憶媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を実行する印刷制御装置および印刷制御方法および記憶媒体に関するものである。

【０００２】

【従来の技術】従来、コンピュータ上で動作するアプリケーションプログラムが、印刷を実行する場合には、印刷開始命令と共に、印刷に必要な各種設定をパラメータとして、プリンタドライバ等の印刷関連処理プログラムに引き渡すのが一般的な方法である。このように印刷に必要な各種設定をパラメータとしてプログラムよりプリンタドライバに渡すことで、プログラムが意図した印刷出力を得られるようになっている。

【０００３】

【発明が解決しようとする課題】しかしながら、意図した印刷出力を得るために指定可能な設定値の種類の範囲はいわゆるどのプリンタでもサポートが可能と思われる用紙の給紙口の指定や両面／片面印刷の指定等、非常に限られた範囲のものであり、プリンタまたはプリンタドライバが提供していても、アプリケーションプログラムより直接指定ができないその他の有用な機能については、プリンタドライバのユーザインタフェースを開いて、そこで指定をする必要があった。

【０００４】このため、企業の各種システムソフトウェアからの印刷出力に際して、プリンタドライバのユーザ

インタフェースを表示したがらないケースでは、プリンタまたはプリンタドライバの持つ機能が十分に利用できないという問題点があった。

【０００５】そのため、昨今ではプリンタまたはプリンタドライバをフルに生かせるように、プリンタやプリンタドライバへの機種に依存する設定についても、アプリケーションから利用可能な特別なインタフェースを設けて、このインタフェースを利用した場合には、プリンタドライバユーザインタフェースで設定する場合と同等な機能の設定が可能となるようなプリンタドライバが出現してきている。

【０００６】こうなると、このインタフェースを介したアプリケーションプログラムからの設定についても、プリンタドライバユーザインタフェースと同等のコンフリクト処理（複数の設定値間での設定の競合状態を解消して、いずれかの設定値の調整を行い、設定値間の整合性を取るための処理）を必要とするので、効率よく、漏れの少ない、そして、ユーザインタフェースで利用されているコンフリクト処理と同等のコンフリクト処理をユーザインタフェース以外の場面でも実現する必要が生じているわけである。

【０００７】本発明は、上記の問題点を解決するためになされたもので、本発明の目的は、独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を記憶しておき、該記憶されたコンフリクト処理ルール設定の設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成し、該自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行することにより、プログラム開発者などが用意するコンフリクト条件を基に補完的に成立するコンフリクト条件を自動生成して、該コンフリクト条件に加えながら、より確かなコンフリクト処理を実現することが可能とするとともに、印刷処理関連プログラムの内部構造体のどのメンバが更新されたのかをアプリケーションが知ることができる印刷制御装置および印刷制御方法および記憶媒体を提供することである。

【０００８】

【課題を解決するための手段】本発明に係る第１の発明は、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置であって、独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を記憶する記憶手段（図１に示す外部メモリ１１に相当）と、前記記憶手段に記憶された複数のコンフリクト処理ルールをまとめたコンフリクト処理ルール設定を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成手段（図３に示すコンフリクトマネージャ３０３，推論エンジン３０２に相当）と、前記生成手段により自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理手段（図３に示すコンフリクトマネージャ３０３，推論エンジン３０２に相当）とを備えるものである。

【０００９】本発明に係る第２の発明は、前記コンフリクト処理手段によりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知手段（図９に示すプリンタドライバ管理部３０６に相当）を備えるものである。

【００１０】本発明に係る第３の発明は、前記印刷処理関連プログラム（図２に示す印刷処理関連プログラム２０４に相当）は、プリンタドライバである。

【００１１】本発明に係る第４の発明は、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置における印刷制御方法であって、記憶手段（図１に示す外部メモリ１１に相当）に記憶される独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成ステップ（図８に示すステップ（５０１）～（５０３））と、前記生成ステップにより自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理ステップ（図８のステップ（５０５）～（５０７））とを備えるものである。

【００１２】本発明に係る第５の発明は、前記コンフリクト処理ステップによりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知ステップ（図示しない）を備えるものである。

【００１３】本発明に係る第６の発明は、前記印刷処理関連プログラム（図２に示す印刷処理関連プログラム２０４に相当）は、プリンタドライバである。

【００１４】本発明に係る第７の発明は、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置に、記憶手段（図１に示す外部メモリ１１に相当）に記憶される独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成ステップ（図８に示すステッ

プ（５０１）～（５０３））と、前記生成ステップにより自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理ステップ（図８のステップ（５０５）～（５０７））とを実行させるためのプログラムを記録媒体にコンピュータが読み取り可能に記録させたものである。

【００１５】本発明に係る第８の発明は、前記コンフリクト処理ステップによりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知ステップ（図示しない）を備えるものである。

【００１６】本発明に係る第９の発明は、前記印刷処理関連プログラム（図２に示す印刷処理関連プログラム２０４に相当）は、プリンタドライバである。

【００１７】

【発明の実施の形態】図１は、本発明の第１実施形態を示す印刷制御装置を適用可能な印刷処理システムの構成を説明するブロック図である。

【００１８】なお、特に断らない限り、本発明の機能が実行されるのであれば、単体の機器であっても、複数の機器からなるシステムであっても、ＬＡＮ，ＷＡＮ等のネットワークを介して接続が為され処理が行われるシステムであっても本発明を適用できることは言うまでもない。

【００１９】図１において、３０００はホストコンピュータで、ＲＯＭ３のプログラム用ＲＯＭあるいは外部メモリ１１に記憶された文書処理プログラム等に基づいて図形、イメージ、文字、表（表計算等を含む）等が混在した文書処理を実行するＣＰＵ１を備え、システムバス４に接続される各デバイスをＣＰＵ１が総括的に制御する。

【００２０】また、このＲＯＭ３のプログラム用ＲＯＭあるいは外部メモリ１１には、ＣＰＵ１の制御プログラムであるオペレーティングシステムプログラム（以下ＯＳ）等を記憶し、ＲＯＭ３のフォント用ＲＯＭあるいは外部メモリ１１には上記文書処理の際に使用するフォントデータ等を記憶し、ＲＯＭ３のデータ用ＲＯＭあるいは外部メモリ１１には上記文書処理等を行う際に使用する各種データを記憶する。

【００２１】２はＲＡＭで、ＣＰＵ１の主メモリ、ワークエリア等として機能する。５はキーボードコントローラ（ＫＢＣ）で、キーボード（ＫＢ）９や不図示のポインティングデバイスからのキー入力を制御する。

【００２２】６はＣＲＴコントローラ（ＣＲＴＣ）で、ＣＲＴディスプレイ（ＣＲＴ）１０の表示を制御する。７はディスクコントローラ（ＤＫＣ）で、ブートプログラム、各種のアプリケーション、フォントデータ、ユーザファイル、編集ファイル、プリンタ制御コマンド生成プログラム（以下プリンタドライバ）等を記憶するハードディスク（ＨＤ）、フロッピー（登録商標）ディスク（ＦＤ）等の外部メモリ１１とのアクセスを制御する。

【００２３】８はプリンタコントローラ（ＰＲＴＣ）で、所定の双方向性インターフェース（インターフェース）２１を介してプリンタ１５００に接続されて、プリンタ１５００との通信制御処理を実行する。なお、ＣＰＵ１は、例えばＲＡＭ２上に設定された表示情報ＲＡＭへのアウトラインフォントの展開（ラスタライズ）処理を実行し、ＣＲＴ１０上でのＷＹＳＩＷＹＧを可能としている。また、ＣＰＵ１は、ＣＲＴ１０上の不図示のマウスカーソル等で指示されたコマンドに基づいて登録された種々のウインドウを開き、種々のデータ処理を実行する。ユーザは印刷を実行する際、印刷の設定に関するウインドウを開き、プリンタの設定や、印刷モードの選択を含むプリンタドライバに対する印刷処理方法の設定を行える。

【００２４】プリンタ１５００において、１２はプリンタＣＰＵ（ＣＰＵ）で、ＲＯＭ１３のプログラム用ＲＯＭに記憶された制御プログラム等あるいは外部メモリ１４に記憶された制御プログラム等に基づいてシステムバス１５に接続される印刷部（プリンタエンジン）１７に出力情報としての画像信号を出力する。

【００２５】また、このＲＯＭ１３のプログラム用ＲＯＭには、ＣＰＵ１２の制御プログラム等を記憶する。ＲＯＭ１３のフォント用ＲＯＭには上記出力情報を生成する際に使用するフォントデータ等を記憶し、ＲＯＭ１３のデータ用ＲＯＭにはハードディスク等の外部メモリ１４がないプリンタの場合には、ホストコンピュータ上で利用される情報等を記憶している。

【００２６】ＣＰＵ１２は入力部１８を介してホストコンピュータとの通信処理が可能となっており、プリンタ内の情報等をホストコンピュータ３０００に通知可能に構成されている。

【００２７】１９は前記ＣＰＵ１２の主メモリ、ワークエリア等として機能するＲＡＭで、図示しない増設ポートに接続されるオプションＲＡＭによりメモリ容量を拡張することができるように構成されている。

【００２８】なお、ＲＡＭ１９は、出力情報展開領域、環境データ格納領域、ＮＶＲＡＭ等に用いられる。前述したハードディスク（ＨＤ）、ＩＣカードやフロッピーディスク（ＦＤ）等の外部メモリ１４は、メモリコントローラ（ＤＫＣ）２０によりアクセスを制御される。外部メモリ１４は、オプションとして接続され、フォントデータ、エミュレーションプログラム、フォームデータ等を記憶する。

【００２９】また、１５０１はた操作パネルで、操作のためのスイッチおよびＬＥＤ表示器等が配されている。また、前述した外部メモリは１個に限らず、少なくとも１個以上備え、内蔵フォントに加えてオプションフォントカード、言語系の異なるプリンタ制御言語を解釈する

プログラムを格納した外部メモリを複数接続できるように構成されていてもよい。さらに、図示しないＮＶＲＡＭを有し、操作パネル１５０１からのプリンタモード設定情報を記憶するようにしてもよい。

【００３０】図２は、図１に示したＲＡＭ３のメモリマップの一例を示す図であり、本実施形態における印刷処理関連プログラムがホストコンピュータ３０００上のＲＡＭ２にロードされ実行可能となった状態のメモリマップを示している。

【００３１】図２において、２０１はアプリケーションで、外部メモリ１１からロードされたものであり、単数または複数のアプリケーションがロードされる。

【００３２】２０２は空きメモリで、ＣＰＵ１のワークとして機能する。２０３は関連データで、アプリケーション２０１の実行に伴う各種の関連データがロードされる。

【００３３】２０４は印刷処理関連プログラムで、外部メモリ１１からロードされる。２０５はＯＳで、アプリケーション２０１等の実行を制御する。２０６はＢＩＯＳで、各種のデバイスの入出力処理を規定する。

【００３４】図３は、図２に示した印刷処理関連プログラムにおけるコンフリクト処理制御部の内部処理の内容を説明するシステム概要図である。

【００３５】図３において、３０１はコンフリクト処理ルールで、コンフリクトマネージャ３０３により図１に示したＲＡＭ２上に読み込まれる。３０２は推論エンジンで、図３に例示してあるように表記されたコンフリクト処理ルール３０１をコンフリクトマネージャ３０３を介して、ＲＡＭ２中に読み込んで補完ルールを生成する。

【００３６】３０４は状態変数リストで、各プリンタ機能名Ａ，Ｂ，Ｃ（図４参照）が推論エンジン３０２により初期化される。コンフリクト処理ルール４０３の中に出現するプリンタ機能名Ａ，Ｂ，Ｃのそれぞれについて同名の状態変数が存在する。

【００３７】３０５は内部構造体で、プリンタ機能名Ａ，Ｂ，Ｃに対応するプリンタドライバ内部構造体４０１のメンバをそれぞれｃＡ，ｃＢ，ｃＣとして管理する。３０６はプリンタドライバ管理部で、コンフリクトマネージャ３０３を呼び出し、状態変数リスト３０４にあるＬａｙｏｕｔの状態変数を更新する。３０７はアプリケーションである。

【００３８】図４は、本発明に係る印刷制御装置におけるコンフリクト処理における各モジュールで扱うデータの関連性を説明する図である。

【００３９】図４において、４０１はプリンタドライバの内部構造体で、図３に示したコンフリクトマネージャ３０３またはプリンタドライバ管理部３０６により読み出しあるいは書き込み可能に構成されている。４０２は状態変数リストで、図３に示した状態変数リストと同等であって、コンフリクトマネージャ３０３により読み出しあるいは書き込み可能に構成されている。なお、図中のＲは読み出しを表し、Ｗは書き込みを表し、Ｒ／Ｗは読み出しあるいは書き込みを表す。

【００４０】図５は、本発明に係る印刷制御装置における補完処理前のコンフリクト処理ルールの例を示す図である。

【００４１】図５において、（１）～（３）は補間処理前のコンフリクト処理ルール３０１の一例を示し、ルール（１）はプリンタドライバ内部構造体４０１のメンバとして存在するＧｒｏｕｐに、ルール（２）はプリンタドライバ内部構造体４０１のメンバとして存在するＳｔａｐｌｅに、ルール（３）はプリンタドライバ内部構造体４０１のメンバとして存在するＬａｙｏｕｔに対応する。

【００４２】図６は、本発明に係る印刷制御装置における補完処理後のコンフリクト処理ルールの例を示す図である。

【００４３】図６において、（１）～（９）は補完処理後のコンフリクト処理ルール（ルール）であり、特に、ルール（６）～（９）における__Ａは推論エンジン３０２で使用される変数を示す。

【００４４】図７は、本発明に係る印刷制御装置における設定変更処理関数の一例を示す図であり、アプリケーションからプリンタドライバへ設定変更処理要求をする場合に利用する関数のインタフェース（Ｉ／Ｆ）の例である。

【００４５】図８は、本発明に係る印刷制御装置におけるデータ処理手順の一例を示すフローチャートであり、本実施形態におけるコンフリクト処理手順に対応する。なお、（５０１）～（５０８）は各ステップを示す。

【００４６】先ず、本実施形態に示す印刷処理関連プログラム（いわゆるプリンタドライバ）の処理は、ＯＳ２０５管理の下で動作するアプリケーション２０１よりプリンタドライバの備える初期化処理が呼び出されると、ＯＳ２０５の管理の下、ＲＡＭ２に図２に示した印刷処理関連プログラム２０４がロードされる。

【００４７】そして、印刷処理関連プログラム２０４がＲＡＭ２にロードされると、プリンタドライバ管理部３０６に備えられたプリンタドライバの初期化処理部が呼び出され、初期化処理が行われる。

【００４８】先ず、図３に示す推論エンジン３０２は、図５に例示してあるように表記されたコンフリクト処理ルール３０１をコンフリクトマネージャ３０３を介して、ＲＡＭ２中に読み込む（５０１）。

【００４９】続いて、ステップ（５０１）で読み込んだコンフリクト処理ルール３０１を元に、２状態値（「ＯＮ」と「ＯＦＦ」）を値として持つルールについては、補完ルールを生成する（５０２）。

【００５０】なお、同じプリンタ機能名について以下の

例のように左辺に「ＯＮまたはＯＦＦ」のどちらかだけが記述された場合、

Ａ（ＯＮ）<-Ｂ（ＯＮ），Ｃ（ＯＦＦ）……（１）

Ａ（ＯＮ）<-Ｄ（Ｖ１）．　　……（２）

Ｂ（ＯＦＦ）<-Ｅ（ＯＦＦ）．　　……（３）

この場合には推論エンジン３０２がＯＮ／ＯＦＦ逆側のルールを以下のように自動生成する。

【００５１】上記ルール（１），（２）に対しては、

Ａ（ＯＦＦ）<-ｎｏｔ　Ａ（ＯＮ）．……（４）

が生成され、上記（３）に対しては、

Ｂ（ＯＮ）<-ｎｏｔ　Ｂ（ＯＦＦ）．……（５）

が生成される。この自動生成されたルール（４），（５）は処理効率上最適化されて下記のルール（４）’，（５）’のように変形されるが、意味は全く同じとなる。

Ａ（ＯＦＦ）<-ｔｒｕｅ．　　……（４）’

Ｂ（ＯＮ）<-ｔｒｕｅ．　　……（５）’

なお、上記例のように逆側が自動生成された場合は、論理として考えた際に、Ａ（ＯＮ）とＡ（ＯＦＦ）は完全に排他な関係にある事を意味している。

【００５２】つまり、Ａ（ＯＮ）とＡ（ＯＦＦ）とでＡについての集合空間が１００％埋め尽くされる事を意味している。Ｂ（ＯＮ）とＢ（ＯＦＦ）も同様である。

【００５３】ユーザがＡ（ＯＮ）とＡ（ＯＦＦ）を混在して記述する場合にはルール（４），（４）’は自動生成されない。従って、この場合には全てのケースを網羅的に記述してＡについての集合空間を１００％埋め尽くすようにする必要がある。

【００５４】続いて、図４に示す状態変数リスト４０２とコンフリクト処理ルール４０３で使用されるプリンタ機能名の値の初期化に関するコンフリクト処理ルールを補完ルールとして自動生成する（ステップ５０３）。

【００５５】コンフリクト処理ルール４０３における全てのプリンタ機能名についてはコンフリクトマネージャ３０３の内部に、図４に示すような状態変数リスト４０２として、その状態変数を持っている。

【００５６】この状態変数の値はプリンタドライバで使用される内部構造体の対応するメンバの値と連動している。全てのプリンタ機能名の状態変数の初期値はその内部構造体３０５のメンバの値が初期値となる。例えば以下のコンフリクト処理ルールを記述した場合、「Ａ（ＯＮ）<-Ｂ（ＯＮ），Ｃ（ＯＦＦ）．」は、図４に示した状態変数リスト４０２に示すようにコンフリクト処理ルール４０３の中に出現するプリンタ機能名Ａ，Ｂ，Ｃのそれぞれについて同名の状態変数が存在している。

【００５７】プリンタ機能名Ａ，Ｂ，Ｃに対応するプリンタドライバ内部構造体４０１のメンバをそれぞれｃＡ，ｃＢ，ｃＣとする。

【００５８】なお、ｉｎｔ　ｃＡの初期値は「０」なので、それに対応する状態変数Ａの値は「ＯＦＦ」となっている。

【００５９】従って、推論エンジン３０２内のプリンタ機能名Ａの状態値の初期値もＯＦＦとなっている。

【００６０】今、推論エンジン３０２でコンフリクトチェックの推論が行われて以下のルール「Ａ（ＯＮ）<-Ｂ（ＯＮ），Ｃ（ＯＦＦ）．」が成立した場合、推論エンジン３０２は左辺のプリンタ機能名Ａの状態変数値を「ＯＮ」に変更する。

【００６１】コンフリクトチェックの推論が終了した後、コンフリクトマネージャ３０３は変更された状態変数の値をプリンタドライバ内部構造体４０１の対応するメンバｉｎｔ　ｃＡに反映させる。

【００６２】つまり、ｉｎｔ　ｃＡは上記ルールが成立した事によって「０」から「１」に変更される。

【００６３】推論エンジン３０２は、推論エンジンに組込まれている組込み関数ｓｔａｔｕｓ（ａ，＿Ｘ）によってプリンタ機能名Ａの状態変数値を推論エンジン３０２で使用される変数＿Ｘに受ける事ができる。推論エンジン３０２は、コンフリクト処理ルールをロードした後、そのコンフリクト処理ルール中に出現する全てのルール名について、次のルールを自動的に生成する。

Ａ（＿Ｘ）<-ｓｔａｔｕｓ（Ａ，＿Ｘ）．

Ｂ（＿Ｘ）<-ｓｔａｔｕｓ（Ｂ，＿Ｘ）．

Ｃ（＿Ｘ）<-ｓｔａｔｕｓ（Ｃ，＿Ｘ）．

これは他に適用するルールが存在しない場合には、プリンタドライバ内部構造体４０１の対応するメンバの値がそのプリンタ機能名の状態値となる事を意味する。

【００６４】プリンタ機能名Ａについては、ルール「Ａ（ＯＮ）<-Ｂ（ＯＮ），Ｃ（ＯＦＦ）．が成り立つのでプリンタ機能名Ａの状態値はこれを受けて［ＯＮ］となる。プリンタ機能名Ｂがもし上記自動生成されたルール以外にルールが存在しないならば、ルール「Ｂ（＿Ｘ）<-ｓｔａｔｕｓ（Ｂ，＿Ｘ）．が適用される。この自動生成ルールは必ず成り立つので、プリンタ機種名Ｂの状態変数の値「ＯＮ」が「＿Ｘ」にユニファイされ、それがプリンタ機種名Ｂの状態値となる。

【００６５】つまり、ユーザ定義ルールが存在しないか、または存在していても成り立つものが無いプリンタ機能については、プリンタドライバ内部構造体４０１の対応するメンバに格納されている値がそのプリンタ機能の状態値という事になる。

【００６６】続いて、その他の初期化処理を行い、処理化処理を終了しアプリケーションに制御を戻す（ステップ５０４）。

【００６７】初期化処理終了後は、プリンタドライバ管理部３０６に備えられている図７に示す設定変更処理関数を呼び出してのアプリケーションからの設定変更処理を繰り返す（ステップ５０５～５０７）。

【００６８】この関数で用いられている引数の中で、特に設定変更処理に関わるパラメータは、ｐＤｅｖＭｏｄ

ｅＩｎＯｕｔｐｕｔと、ｐＩｔｅｍＰａｒａｍ，ｄｗＩｔｅｍＩＤである。それぞれのパラメータには以下のような情報がセットされる。

【００６９】パラメータｄｗＩｔｅｍＩＤには、変更を行いたいのはプリンタドライバ内部構造体４０１のどのメンバなのかを指定する。パラメータｐＩｔｅｍＰａｒａｍには、パラメータｄｗＩｔｅｍＩＤで示したメンバのパラメータを指定する。また、パラメータｐＤｅｖＭｏｄｅＩｎＯｕｔｐｕｔには、プリンタドライバ内部構造体４０１へのポインタを指定する。

【００７０】例えば、アプリケーションからの設定変更要求がＰｒｉｎｔ　Ｓｔｙｌｅを１－ｓｉｄｅｄ　ＰｒｉｎｔｉｎｎｇからＢｏｏｋｌｅｔ　Ｐｒｉｎｔｉｎｎｇに変更するものであった場合を例にとると、上記の各々のパラメータには、パラメータｄｗＩｔｅｍＩＤには、「ＤＭ＿ＩＴＥＭ＿ＬＡＹＯＵＴ」が、パラメータｐＩｔｅｍＰａｒａｍには、「ＤＭ＿ＰＡＲＡＭ＿ＢＯＯＫＬＥＴ」が、パラメータｐＤｅｖＭｏｄｅＩｎＯｕｔｐｕｔには、「ｐＤｅｖＭｏｄｅ」がセットされた上で、ＤｏｃｕｍｅｎｔＰｒｏｐｅｒｔｉｅｓＥｘ（）が呼び出される。

【００７１】すると、図５のステップ（５０５）～（５０７）に示すプリンタドライバの設定変更処理が開始されることとなる。

【００７２】まず、ステップ（５０５）で実行されるコンフリクト処理ルールの適用について説明する。

【００７３】上記の例を用いて説明すると、コンフリクト処理ルールの適用は、図５に示すコンフリクト処理ルール３０１の一部として読み込まれたルールと、ステップ（５０２）にて生成された補完ルールの一部である図６に示した（４），（５）とステップ（５０３）にて生成されたルールの一部である図６に示した（６）～（９）に対して行われる。

【００７４】プリンタドライバ内部構造体４０１のメンバとして存在するＣｏｌｌａｔｅ，Ｇｒｏｕｐ，Ｓｔａｐｌｅ，Ｌａｙｏｕｔの各メンバのコンフリクト処理ルール適用前の値は、Ｃｏｌｌａｔｅは「ＯＦＦ」で、Ｇｒｏｕｐが「ＯＮ」で、Ｓｔａｐｌｅが「ＯＦＦ」で、Ｌａｙｏｕｔが「１－Ｓｉｄｅ」となる。

【００７５】そして、アプリケーションからの変更要求がＬａｙｏｕｔをＬａｙｏｕｔからＢｏｏｋｌｅｔに変更するものであるので、Ｌａｙｏｕｔのメンバの内容は、Ｃｏｌｌａｔｅは「ＯＦＦ」で、Ｇｒｏｕｐが「ＯＮ」で、Ｓｔａｐｌｅが「ＯＦＦ」で、Ｌａｙｏｕｔが「Ｂｏｏｋｌｅｔ」となる。

【００７６】次に、プリンタドライバ管理部３０６はコンフリクトマネージャ３０３を呼び出し、状態変数リスト３０４にあるＬａｙｏｕｔの状態変数が更新され、続いて、推論エンジン３０２がコールされて、コンフリクト処理ルール３０１の適用が始まる。

【００７７】まず、図６に示したルール（６）～（９）が適用され、推論エンジン３０２内の各プリンタ機能名が状態変数リスト３０２の各メンバの持つ値で初期化される。続いて、図６に示したルール（３）が適用され、Ｇｒｏｕｐの値は「ＯＮ」から「ＯＦＦ」へと変化する。

【００７８】これにより、Ｌａｙｏｕｔのメンバの内容は、Ｃｏｌｌａｔｅは「ＯＦＦ」で、Ｇｒｏｕｐが「ＯＦＦ」で、Ｓｔａｐｌｅが「ＯＦＦ」で、Ｌａｙｏｕｔが「Ｂｏｏｋｌｅｔ」となる。

【００７９】さらに、図６に示すルール（４）が適用され、Ｃｏｌｌａｔｅは「ＯＦＦ」から「ＯＮ」へと変化する。

【００８０】これにより、Ｌａｙｏｕｔのメンバの内容は、Ｃｏｌｌａｔｅは「ＯＮ」で、Ｇｒｏｕｐが「ＯＦＦ」で、Ｓｔａｐｌｅが［ＯＦＦ」で、Ｌａｙｏｕｔが「Ｂｏｏｋｌｅｔ」となる。

【００８１】他に、適用されるルールが存在しないので、以上で推論エンジン３０２でのコンフリクト処理ルールの適用が終了する。

【００８２】次のステップでは、コンフリクトマネージャ３０３が上記の最終状態を元に状態変数リスト３０４の更新（ステップ５０６）と、プリンタドライバ内部構造体の更新（ステップ５０７）を行う。

【００８３】本来であればアプリケーション自身の手により、このＬａｙｏｕｔの設定関連するメンバであるＧｒｏｕｐとＣｏｌｌａｔｅのメンバの値を調整する必要があるが、ＤｏｃｕｍｅｎｔＰｒｏｐｅｔｉｔｅｓＥｘを利用することで、この例の場合、アプリケーションは、Ｌａｙｏｕｔを「１－Ｓｉｄｅ」から「Ｂｏｏｋｌｅｔ」に変更する処理をプリンタドライバに依頼することによって、Ｇｒｏｕｐが「ＯＮ」から「ＯＦＦ」に、Ｃｏｌｌａｔｅが「ＯＦＦ」から「ＯＮ」に更新されたプリンタドライバ内部構造体３０１を受け取ることができる。

【００８４】この後、ここで取得したプリンタドライバ内部構造体３０１をパラメータとして実際の印刷を開始すれば、アプリケーションが所望する印刷結果が簡便に得られることとなる。

【００８５】以上の処理は、アプリケーションによってプリンタドライバ管理部３０６に備えられた終了処理が呼び出されるまで、繰り返し実行される。アプリケーションよりプリンタドライバの終了処理が呼び出されると空きメモリ２０２に生成したコンフリクト処理ルール情報等を消去すると共にその他の終了処理を行う（ステップ５０８）。

【００８６】これにより処理は全て終了し、本実施形態における印刷処理関連プログラム２０４の処理も終了し、ＲＡＭ２からはＯＳ２０５の機能により消去される。

【００８７】なお、本実施形態においては、本印刷処理関連プログラム２０４を記録する媒体を外部メモリとしているが、外部メモリとしては、ＦＤ、ＨＤドライブ、ＣＤ－ＲＯＭやＩＣメモリカード等であってもよい。更に、本印刷プログラム単独、もしくはＯＳその他のホストコンピュータ上で動作するプログラムと共にＲＯＭ３に記録しておき、これをメモリマップの一部となすように構成し、直接ＣＰＵ１で実行することも可能である。

【００８８】〔第２実施形態〕上記第１実施形態においては、プリンタドライバ内部構造体４０１の設定内容をコンフリクト処理を実施することで、自動的に最適な値に更新することとなるが、さらに、図９、図１０に示すとおり、コンフリクトマネージャ３０３にコンフリクト処理結果リスト９０１を加え、状態変数リスト４０２が更新されるのと同時に、その中に状態変数の変更履歴を書きこむことで、コンフリクト処理を実行することにより、プリンタドライバ内部構造体のどのメンバが更新されたのかをアプリケーションが知ることができるようになる。

【００８９】図９は、本発明の第２実施形態を示す印刷制御装置におけるコンフリクト処理制御部の内部処理の内容を説明するシステム概要図であり、図３と同一のものには同一の符号を付してある。

【００９０】図１０は、本発明に係る印刷制御装置におけるコンフリクト処理における各モジュールで扱うデータの関連性を説明する図であり、図４と同一のものには同一の符号を付してある。

【００９１】図１１は、本発明に係る印刷制御装置における他の設定変更処理関数の一例を示す図である。

【００９２】図９に示すコンフリクト処理結果リストの例は、プリンタドライバ内部構造体４０１のメンバの内、コンフリクト処理ルールにて更新がされたメンバは１個ということが、ＲｅｓｕｌｔＮｕｍに保存され、そして、更新されたメンバのリストが後続のＭｏｄｉｆｉｅｄ［］に保存され、この場合はＲｅｓｕｌｔＮｕｍが１であるので、リスト内のメンバも１つで、ＩＤ＿ｃａ、すなわち、プリンタドライバ内部構造体４０１のｃＡが更新されたことを表現している。

【００９３】アプリケーション３０７は、このコンフリクト処理結果リストへのポインタを図１１に示すＤｏｃｕｍｅＰｒｏｐｅｒｔｉｅｓＥｘ関数の戻り値として受け取ることで、その情報にアクセスできるようになる。

【００９４】上記実施形態を例にとれば、コンフリクト処理結果リスト９０１の内容としては、ＩＤ＿ＧｒｕｏｐとＩＤ＿Ｃｏｌｌａｔｅが含まれる。アプリケーション３０７は、このコンフリクト処理結果リスト９０１の内容を確認して、実際にその値が格納されているプリンタドライバ内部構造体４０１の該当するメンバ、Ｇｒｏｕｐ，Ｃｏｌｌａｔｅの内容を確認し、その設定内容がアプリケーション３０７が望まない設定変更で合った場合には、Ｌａｙｏｕｔを「１－Ｓｉｄｅ」から「Ｂｏｏｋｌｅｔ」への設定の変更を取り消すことも可能である。

【００９５】上記各実施形態によれば、プログラム開発者などが用意するコンフリクト条件を補完するコンフリクト条件を自動生成するので、より品質の高いコンフリクト処理を実現できる。

【００９６】さらに、コンフリクト処理が実行されることによって、設定が変更されたかどうかの情報を各設定値について取得できるので、この情報に基づいた処理、例えば、他の設定項目に影響を与える元となった設定変更の取り消しやユーザへのメッセージ出力等を容易に実装できる。

【００９７】以下、図１２に示すメモリマップを参照して本発明に係る印刷制御装置を適用可能な印刷システムで読み出し可能なデータ処理プログラムの構成について説明する。

【００９８】図１２は、本発明に係る印刷制御装置を適用可能な印刷システムで読み出し可能な各種データ処理プログラムを格納する記憶媒体のメモリマップを説明する図である。

【００９９】なお、特に図示しないが、記憶媒体に記憶されるプログラム群を管理する情報、例えばバージョン情報，作成者等も記憶され、かつ、プログラム読み出し側のＯＳ等に依存する情報、例えばプログラムを識別表示するアイコン等も記憶される場合もある。

【０１００】さらに、各種プログラムに従属するデータも上記ディレクトリに管理されている。また、各種プログラムをコンピュータにインストールするためのプログラムや、インストールするプログラムが圧縮されている場合に、解凍するプログラム等も記憶される場合もある。

【０１０１】本実施形態における図８に示す機能が外部からインストールされるプログラムによって、ホストコンピュータにより遂行されていてもよい。そして、その場合、ＣＤ－ＲＯＭやフラッシュメモリやＦＤ等の記憶媒体により、あるいはネットワークを介して外部の記憶媒体から、プログラムを含む情報群を出力装置に供給される場合でも本発明は適用されるものである。

【０１０２】以上のように、前述した実施形態の機能を実現するソフトウエアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読出し実行することによっても、本発明の目的が達成されることは言うまでもない。

【０１０３】この場合、記憶媒体から読み出されたプログラムコード自体が本発明の新規な機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【０１０４】プログラムコードを供給するための記憶媒体としては、例えば、フロッピーディスク，ハードディスク，光ディスク，光磁気ディスク，ＣＤ－ＲＯＭ，ＣＤ－Ｒ，磁気テープ，不揮発性のメモリカード，ＲＯＭ，ＥＥＰＲＯＭ等を用いることができる。

【０１０５】また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているＯＳ（オペレーティングシステム）等が実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【０１０６】さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるＣＰＵ等が実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【０１０７】

【発明の効果】以上説明したように、本発明に係る印刷制御装置によれば、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置であって、独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を記憶する記憶手段と、前記記憶手段に記憶された複数のコンフリクト処理ルールをまとめたコンフリクト処理ルール設定を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成手段と、前記生成手段により自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理手段とを備えるので、プログラム開発者などが用意するコンフリクト条件を基に補完的に成立するコンフリクト条件を自動生成し、前記のコンフリクト条件に加えることが可能となり、より確かなコンフリクト処理を実現することが可能なる。

【０１０８】また、前記コンフリクト処理手段によりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知手段を備えるので、印刷処理関連プログラムの内部構造体のどのメンバが更新されたのかをアプリケーションが知ることができるようになる。

【０１０９】本発明に係る印刷制御方法によれば、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御方法であって、記憶手段に記憶される独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成ステップと、前記生成ステップにより自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理ステップとを備えるので、プログラム開発者などが用意するコンフリクト条件を基に補完的に成立するコンフリクト条件を自動生成し、前記のコンフリクト条件に加えることが可能となり、より確かなコンフリクト処理を実現することが可能なる。

【０１１０】また、前記コンフリクト処理ステップによりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知ステップを備えるので、印刷処理関連プログラムの内部構造体のどのメンバが更新されたのかをアプリケーションが知ることができるようになる。

【０１１１】本発明に係る記憶媒体によれば、アプリケーションから印刷装置を制御する印刷処理関連プログラムに対して引き渡される異なる複数の設定値間のコンフリクト処理を行う印刷制御装置に、記憶手段に記憶される独立した２つ以上の設定値の間で成立する依存関係を示す複数のコンフリクト処理ルールをまとめたコンフリクト処理ルールの設定値を読み込み、該読み込んだコンフリクト処理ルールの設定値を基にして、補完的に成立するコンフリクト処理ルールを自動生成する生成ステップと、前記生成ステップにより自動生成されたコンフリクト処理ルールを用いて、前記印刷装置に対する印刷実行に必要な各種設定に関わるコンフリクト処理を実行するコンフリクト処理ステップとを実行させるためのプログラムを記録媒体にコンピュータが読み取り可能に記録させたので、プログラム開発者などが用意するコンフリクト条件を基に補完的に成立するコンフリクト条件を自動生成し、前記のコンフリクト条件に加えることが可能となり、より確かなコンフリクト処理を実現することが可能なる。

【０１１２】また、前記コンフリクト処理ステップによりコンフリクト処理を実施した結果、前記コンフリクト処理ルールの適用により変更がなされた設定を通知する通知ステップを備えるので、印刷処理関連プログラムの内部構造体のどのメンバが更新されたのかをアプリケーションが知ることができるようになる。

【図面の簡単な説明】

【図１】本発明の第１実施形態を示す印刷制御装置を適用可能な印刷処理システムの構成を説明するブロック図である。

【図２】図１に示したＲＡＭのメモリマップの一例を示す図である。

【図３】図２に示した印刷処理関連プログラムにおけるコンフリクト処理制御部の内部処理の内容を説明するシステム概要図である。

【図４】本発明に係る印刷制御装置におけるコンフリクト処理における各モジュールで扱うデータの関連性を説明する図である。

【図５】本発明に係る印刷制御装置における補完処理前のコンフリクト処理ルールの例を示す図である。

【図６】本発明に係る印刷制御装置における補完処理後のコンフリクト処理ルールの例を示す図である。

【図７】本発明に係る印刷制御装置における設定変更処理関数の一例を示す図である。

【図８】本発明に係る印刷制御装置におけるデータ処理手順の一例を示すフローチャートである。

【図９】本発明の第２実施形態を示す印刷制御装置におけるコンフリクト処理制御部の内部処理の内容を説明するシステム概要図である。

【図１０】本発明に係る印刷制御装置におけるコンフリクト処理における各モジュールで扱うデータの関連性を説明する図である。

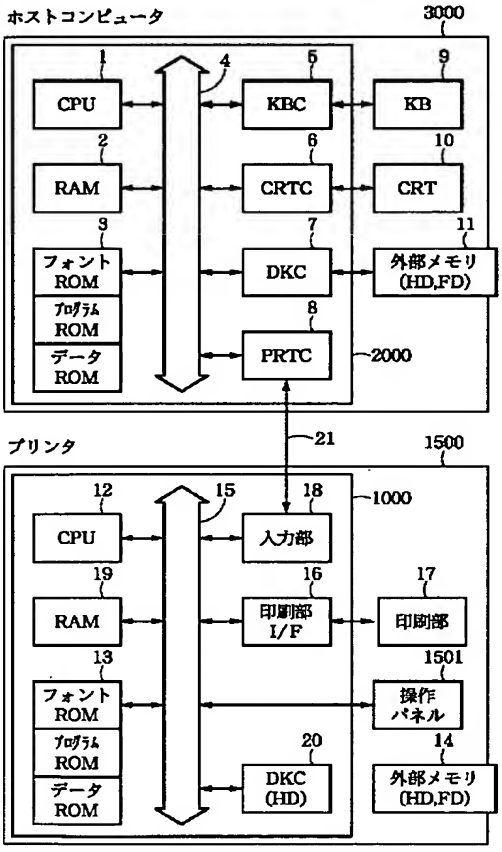
【図１１】本発明に係る印刷制御装置における他の設定変更処理関数の一例を示す図である。

【図１２】本発明に係る印刷制御装置を適用可能な印刷システムで読み出し可能な各種データ処理プログラムを格納する記憶媒体のメモリマップを説明する図である。

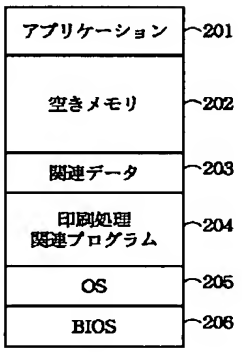
【符号の説明】

１　ＣＰＵ
２　ＲＡＭ
３　ＲＯＭ
４　システムバス
１２　ＣＰＵ
１３　ＲＯＭ
１９　ＲＡＭ
３０００　ホストコンピュータ

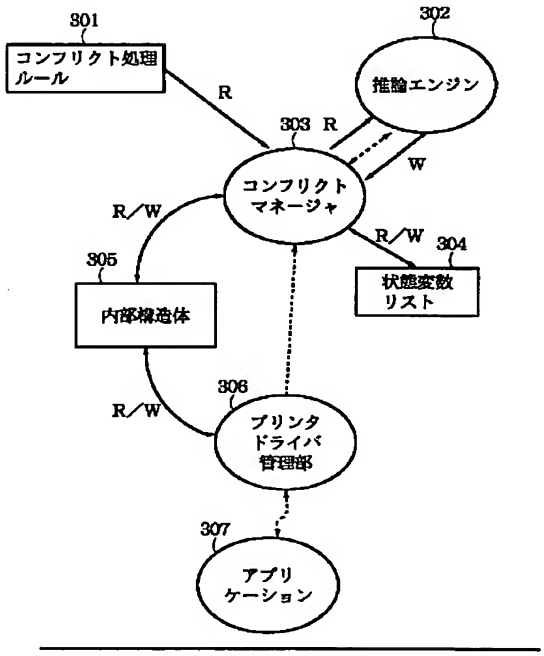
【図１】



【図２】



【図３】

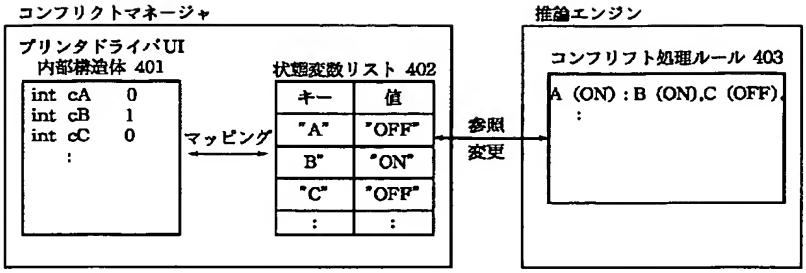


【図５】

301

Collate(OFF)<-Group(ON)　(1)
Collate(OFF)<-Staple(ON)　(2)
Collate(OFF)<-Layout(BOOKLET)　(3)

【図４】



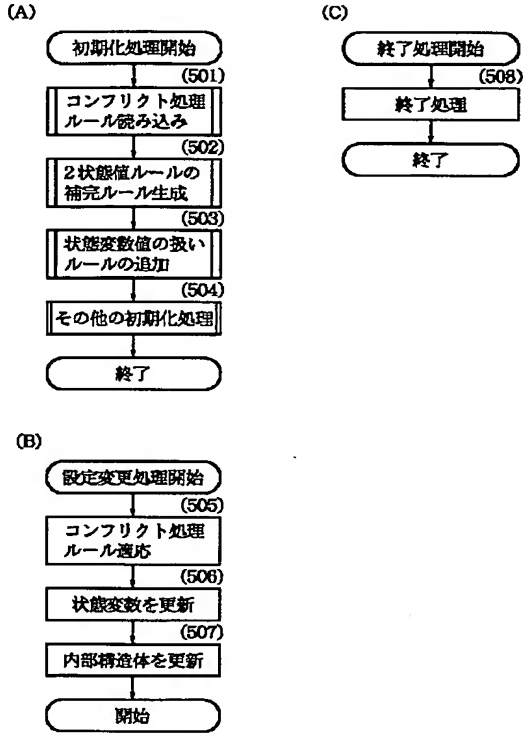
【図６】

```
Collate(OFF)<-Group(ON)                    (1)
Collate(OFF)<-Staple(ON)                   (2)
Collate(OFF)<-Layout(BOOKLET)              (3)

Collate(ON)<-true.                         (4)
Group(ON)<-true.                           (5)

Collate(_X)<-status(Collate,_X).           (6)
Group(_X)<-status(Group,_X).               (7)
Layout(_X)<-status(Layout,_X).             (8)
Staple(_X)<-status(Staple,_X).             (9)
```

【図７】

```
設定変更処理関数
LONG DocumentPropertiesEX(
    HWND hWnd,                  //handle to parent window
    HANDLE hPrinter,            //handle to printer object
    LPTSTR pDeviceName          //device name
    PDEVMODE pDevModeInOutput   //original device mode &
                                //modified device mode
    DWORD dwItemID              //indicate ItemID
    LPVOID pItemParam           //Parameter indicated by ItemID
);
```

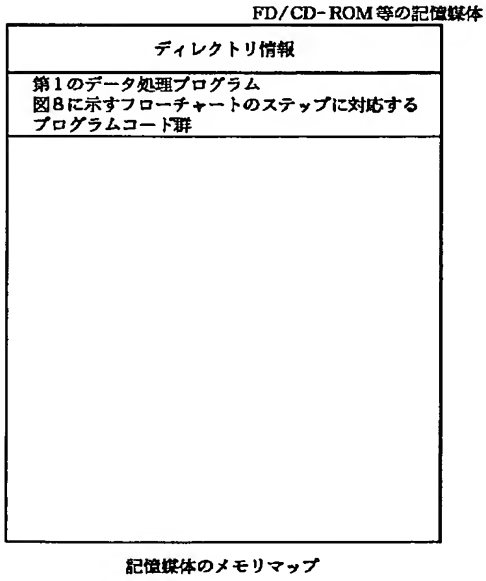
【図８】



【図１１】

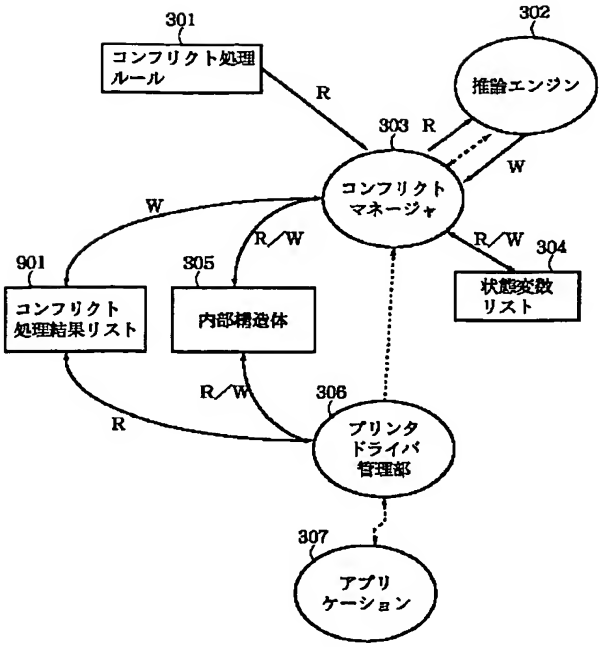
```
設定変更処理関数
LPRESULTLLIST DocumentPropertiesEX(
    HWND hWnd,                  //handle to parent window
    HANDLE hPrinter,            //handle to printer object
    LPTSTR pDeviceName          //device name
    PDEVMODE pDevModeInOutput   //original device mode &
                                //modified device mode
    DWORD dwItemID              //indicate ItemID
    LPVOID pItemParam           //Parameter indicated by ItemID
);
```

【図１２】

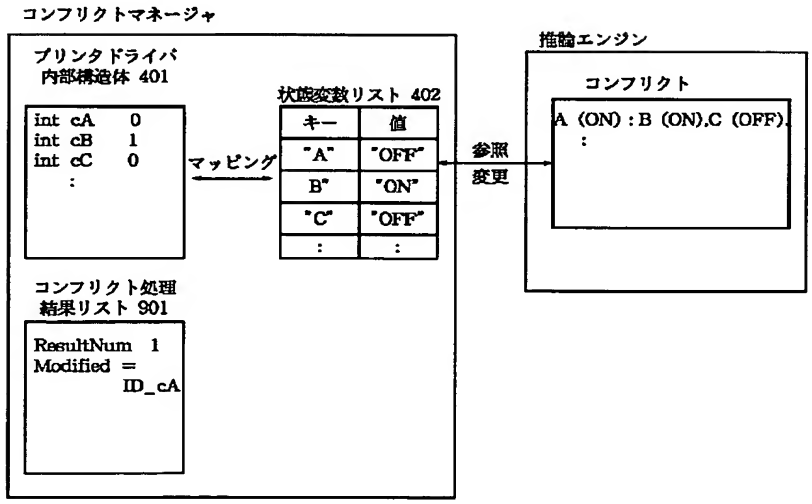


記憶媒体のメモリマップ

【図９】


301
コンフリクト処理ルール
302
推論エンジン
R
303 R
W
コンフリクトマネージャ
W
R／W
R／W
304
状態変数リスト
901
コンフリクト処理結果リスト
305
内部構造体
R／W
306
R
プリンタドライバ管理部
307
アプリケーション

【図１０】


コンフリクトマネージャ
プリンタドライバ内部構造体 401
int cA 0
int cB 1
int cC 0
:
マッピング
状態変数リスト 402
キー
値
"A"
"OFF"
B"
"ON"
"C"
"OFF"
:
:
参照
変更
推論エンジン
コンフリクト
A (ON) : B (ON),C (OFF),
:
コンフリクト処理結果リスト 901
ResultNum 1
Modified =
ID_cA